**Amendments to the Claims:**

This listing of the claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. **(Currently amended)** A method for media streaming, comprising:

receiving a request from a client to a server via a network in accordance with a Hypertext Transfer Protocol (HTTP) to stream a certain portion of a media file of a given type;

passing the request to a servlet running in conjunction with the server, wherein the servlet is a platform-independent class that is compiled to platform-neutral bytecode and is loaded dynamically into and run by the server;

in response to the request, activating a first servlet class instance of the servlet to parse the request in order to identify the portion of the media file requested by the client and to identify a format of the media file;

instantiating, using the first servlet class instance, a second servlet class, which inherits an interface from an abstract parser class and is specified to handle the identified format of the media file;

parsing the ~~request~~ media file using the second servlet class to select~~, responsively to the request,~~ elements of the media file corresponding to the ~~requested~~ identified portion to be transferred to the client; and

streaming the ~~identified~~ selected elements from the server to the client as a HTTP response.

2. **(Currently amended)** A method according to claim 1, wherein activating the first servlet class comprises parsing the request ~~comprises determining~~ to determine a processing action to be applied to the elements of the media file, and wherein streaming the identified elements comprises applying the processing action to the elements.

3. **(Original)** A method according to claim 2, wherein parsing the request comprises determining a parameter applicable to the processing action, and wherein applying the processing action comprises processing the elements of the media file responsive to the parameter.

4. **(Currently amended)** A method according to claim 3, wherein determining the parameter comprises determining a limitation on a media playing capability of the client, and wherein the processing action comprises modifying the ~~identified~~ selected elements in response to the limitation.

5. **(Currently amended)** A method according to claim 4, wherein determining the limitation comprises identifying a network bandwidth, and wherein modifying the ~~identified~~ selected elements in response to the limitation comprises altering the elements responsive to the network bandwidth.

6. **(Currently amended)** A method according to claim 4, wherein determining the limitation comprises determining a resource level provided by the client, and wherein modifying the ~~identified~~ selected elements comprises ~~selecting~~ choosing the ~~identified~~ selected elements responsive to the resource level.

7. **(Original)** A method according to claim 2, wherein applying the processing action comprises transcoding at least one of the elements of the media file into a desired media format.

8. **(Canceled)**

9. **(Currently amended)** A method according to claim 1, wherein the elements of the media file comprise an ordered sequence of frames, and wherein parsing the ~~request~~ media file comprises selecting a segment of the sequence of the frames.

10. **(Currently amended)** A method according to claim 1, wherein the elements of the media file comprises a plurality of media tracks temporally juxtaposed in parallel, and wherein ~~selecting the elements~~ parsing the media file comprises selecting one or more of the tracks.

11. **(Currently amended)** Apparatus for media streaming, comprising a server which is arranged to receive a request from a client via a network in accordance with a Hypertext Transfer Protocol (HTTP) to stream a certain portion of a media file of a given type, and which is further arranged to run a servlet and to pass the request to the servlet, wherein the servlet is a platform-independent class that is compiled to platform-neutral bytecode and is loaded dynamically into and run by the_server,

wherein the server is configured, in response to the request, to activate a first servlet class instance of the servlet to parse the request in order to identify the portion of the media file requested by the client and to identify a format of the media file, and to instantiate, using the first servlet class instance, a second servlet class, which inherits an interface from an abstract parser class and is specified to handle the identified format of the media file, and to parse the ~~request~~ media file using the second servlet class to select~~, responsively to the request,~~ elements of the media file corresponding to the ~~requested~~ identified portion to be transferred to the client, and to stream the ~~identified~~ selected elements from the server to the client as a HTTP response.

12. **(Original)** Apparatus according to claim 11, wherein the server is arranged to use the servlet to parse the request so as to determine a processing action to be applied to the elements of the media file, and to apply the processing action to the elements.

13. **(Original)** Apparatus according to claim 12, wherein the server is arranged to use the servlet to determine a parameter applicable to the processing action, and to apply the processing action based on the parameter.

14. **(Currently amended)** Apparatus according to claim 13, wherein the parameter is indicative of a limitation on a media playing capability of the client, and wherein the server is arranged to apply the processing action so as to modify the ~~identified~~ selected elements in response to the limitation.

15. **(Currently amended)** Apparatus according to claim 14, wherein the limitation applies to a network bandwidth, and wherein the server is arranged to use the servlet to modify the ~~identified~~ selected elements in response to the network bandwidth.

16. **(Currently amended)** Apparatus according to claim 14, wherein the limitation applies to a resource level provided by the client, and wherein the server is arranged to use the servlet to ~~select~~ choose the ~~identified~~ selected elements in response to the resource level.

17. **(Original)** A method according to claim 13, wherein the processing action comprises transcoding at least one of the elements of the media file into a desired media format.

18. **(Canceled)**

19. **(Previously presented)** Apparatus according to claim 11, wherein the elements of the media file comprise an ordered sequence of frames, and wherein the server is arranged to use the servlet to select a segment of the sequence of the frames responsive to the request.

20. **(Previously presented)** Apparatus according to claim 11, wherein the elements of the media file comprises a plurality of media tracks temporally juxtaposed in parallel, and wherein the server is arranged to use the servlet to select one or more of the tracks responsive to the request.

21. **(Original)** Apparatus according to claim 11, wherein the server comprises a cluster of servers, arranged so that the HTTP request is handled by one of the servers in the cluster, and the servlet is run on a different one of the servers in the cluster.

22. **(Currently amended)** A computer software product for media streaming, comprising a computer-readable medium in which program instructions are stored, which instructions, when read by a computer, cause the computer to receive a request from a client via a network in accordance with a Hypertext Transfer Protocol (HTTP) to stream a certain portion of a media file of a given type, and which is further arranged to run a servlet and to pass the request to the servlet, wherein the servlet is a platform-independent class that is compiled to platform-neutral bytecode and is loaded dynamically into and run by the server,

wherein the instructions cause the computer, in response to the request, to activate a first servlet class instance of the servlet to parse the request in order to identify the portion of the media file requested by the client and to identify a format of the media file, and to instantiate, using the first servlet class instance, a second servlet class, which inherits an interface from an abstract parser class and is specified to handle the identified format of the media file, and to parse the ~~request~~ media file using the second servlet class to select~~, responsively to the request,~~ elements of the media file corresponding to the ~~requested~~ identified portion to be transferred to the client, and to stream the ~~identified~~ selected elements from the server to the client as a HTTP response.

23. **(Original)** A product according to claim 22, wherein the instructions cause the computer to use the servlet to parse the request so as to determine a processing action to be applied to the elements of the media file, and to stream the identified elements by applying the processing action to the elements.

24. **(Original)** A product according to claim 23, wherein the instructions cause the computer to use the servlet to determine a parameter applicable to the processing action, and to apply the processing action based on the parameter.

25. **(Currently amended)** A product according to claim 24, wherein the parameter is indicative of a limitation on a media playing capability of the client, and wherein the instructions cause the computer to apply the processing action so as to modify the ~~identified~~ selected elements in response to the limitation.

26. **(Currently amended)** A product according to claim 25, wherein the limitation applies to a network bandwidth, and wherein the instructions cause the computer to use the servlet to modify the ~~identified~~ selected elements in response to the network bandwidth.

27. **(Currently amended)** A product according to claim 25, wherein the limitation applies to a resource level provided by the client, and wherein the instructions cause the computer to use the servlet to ~~select~~ choose the ~~identified~~ selected elements in response to the resource level.

28. **(Original)** A product according to claim 24, wherein the processing action comprises transcoding at least one of the elements of the media file into a desired media format.

29. **(Canceled)**

30. **(Previously presented)** A product according to claim 22, wherein the elements of the media file comprise an ordered sequence of frames, and wherein the instructions cause the computer to use the servlet to select a segment of the sequence of the frames responsive to the request.

31. **(Previously presented)** A product according to claim 22, wherein the elements of the media file comprises a plurality of media tracks temporally juxtaposed in parallel, and wherein the instructions cause the computer to use the servlet to select one or more of the tracks.

32. **(Original)** A product according to claim 22, wherein the servlet comprises a subset of the instructions, and the subset of the instructions comprises instructions written in a platform-independent, object-oriented computer language.